Unidad 7- Trabajando con base de datos (parte 1)

Para realizar una conexión con base de datos necesitamos nuestro entorno instalado (se recomienda utilizar xampp y activar la base de datos mysql)

En este curso trabajaremos con mysql para la conexión a la misma. En PHP tenemos dos conectores distintos a la hora de operar: mysqli y PDO
Se recomienda el uso de PDO por su portabilidad a otros sistemas de gestión de base de datos.

Conexión a una base de datos

Para conectarnos a una base de datos necesitamos conocer los datos de nuestro servidor de base de datos y de donde está alojada. Si usamos una base de datos local, usaremos como dirección del servidor “localhost”, en caso contrario pondremos la ip o nombre del dominio asociado a la BD.
El nombre de usuario y contraseña deben ser un usuario existente en el sistema de gestión de la base de datos.

```php
<?php
    $servidor = "localhost";
    $usuario = "jperez";
    $password = "contra123";

    try {
      $conn = new PDO("mysql:host=$servidor;dbname=prueba", $usuario, $password);
      $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
      echo "Conectado";
    } catch(PDOException $e) {
      echo "Fallo la conexion: " . $e->getMessage();
    }
```

Como PDO es una clase como vimos en la unidad anterior, vamos a instanciar un objeto de la misma, utilizando su constructor parametrizado.
En la siguiente línea definiremos los atributos para que PDO lance una excepción cada vez que ocurra un error (que lo podremos capturar con try/catch)
Si todo sale bien, saldrá el mensaje “Conectado”. En caso contrario, saldrá el mensaje “Fallo la conexión”, seguido del error ocurrido.
Una vez invocada esta conexión recuerde que tiene que cerrarla al finalizar la conexión (con $conn = null).

Insertar un dato en una tabla

En este punto se da por sobreentendido que sabe cómo manejar una base de datos desde mysql así que suponiendo que tenemos una base de datos llamada “prueba” con una tabla “persona” que posee los campos “ci” de tipo int(8), “nombre” de tipo varchar(25) y apellido de tipo varchar(25).
A partir de estos supuestos crearemos un registro en esa tabla

```php
<?php
    $servidor = "localhost";
    $usuario = "jperez";
    $password = "contra123";
    $base = "prueba";

    try {
      $conn = new PDO("mysql:host=$servidor;dbname=$base", $usuario, $password);
      $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

      $sql = "INSERT INTO persona (ci, nombre, apellido)
      VALUES (12345678, 'Juan', 'Perez')";
      $conn->exec($sql);
      echo "Se creo el registro en la base de datos";
    } catch(PDOException $e) {
      echo $sql . "<br>" . $e->getMessage();
    }

    $conn = null;
```

La diferencia con el ejemplo anterior es que agregamos el statement de sql “INSERT”. Al no tener ningún retorno la instrucción “INSERT” usaremos la función exec()
Si todo sale bien saldrá en mensaje correspondiente y al finalizar, cerraremos la conexión a la base de datos. En caso contrario se nos mostrará el mensaje de error

Obtener datos de una tabla

Para obtener datos de una tabla lo que cambiará es el statement

```php
<?php
    $servidor = "192.168.56.104";
    $usuario = "juan";
    $password = "Juan123";
    $base = "prueba";

    try {
      $conn = new PDO("mysql:host=$servidor;dbname=$base", $usuario, $password);
      $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

      $stmt = $conn->prepare("SELECT * FROM persona");
      $stmt->execute();

      $result = $stmt->setFetchMode(PDO::FETCH_ASSOC);
      foreach($stmt->fetchAll() as $fila) {
          foreach($fila as $campo => $valor) {
              echo "$campo: $valor<br>";
          }
          echo "<br>";
      }
    } catch(PDOException $e) {
      echo $sql . "<br>" . $e->getMessage();
    }

    $conn = null;
```

Una aclaración correspondiente a esta sintaxis es que la misma no es segura a la hora de realizarla en un ambiente de producción por lo que veremos en la siguiente parte de la unidad las características de seguridad que debe poseer el statement para evitar vulnerabilidades.